UT 6: Gestión y Almacenamiento de información en formatos XML:

DIGI TECH

Málaga 2023
www.digitechfp.com

Alberto Ruiz Rodriguez

INDICE

1. Sistemas de almacenamiento de información.

XML se usa como formato intercambio de información → Lenguajes que se usa xpath y xquery

XPATH : se usa para buscar y navegar en un documento XML: Filtra nodosm ejemplo nodo de tipo elementos <producto> en el caso de los valores como en el nodo de precio <precio> 200</precio>, 200 es un nodo hijo de precios.

XQUERY : Es un lenguaje de consultas, para realizar búsquedas avanzadas en XML

Ejemplo:

```
for $producto in //producto
where $producto/precio > 1000
return concat($producto/nombreproducto/text(), " - ", $producto/fabricante/text()).
```

En que casos usar bases de datos XML: bases de datos tipo médico, fabricación, tiendas online, etc.

SAI: sistemas de almacenamiento de información

En ficheros

No es aconsejable. No se garantiza:

Concurrencia: muchas transacciones al mismo tiempo.

Integridad de de atomicidad

Propiedad de una transacción

Seguridad: de los datos

En base de datos XML

En realidad es una base de datos relacional que convierte los documentos XML en un esquema relacional:

En base de datos nativas XML

Su modelo interno se basa en XML:

MarkLogic, BaseX, eXistdb, sedna Native XML Database System

Que diferencia hay entre base de datos nativas XML y NO nativas XML

La principal diferencia entre las bases de datos nativas y no nativas XML es que las primeras están diseñadas específicamente para manejar datos en formato XML de manera más eficiente y con mayor funcionalidad, mientras que las segundas pueden ser capaces de almacenar y procesar datos XML, pero no están optimizadas para ello.

2. Inserción y extracción de información en XML.

IMPORTXML → EN GOOGLE SHEET

XMLIMPORT WEBSERVICE → EN EXCEL

WEBSERVICE → EN LIBREOFFICE

IMPORTXML en GOOGLE SHEET

Importa datos de varios tipos de datos estructurados, incluidos XML, HTML, CSV, TSV y feeds XML RSS y ATOM

Ejemplo de uso

IMPORTXML("https://en.wikipedia.org/wiki/Moon_landing"; "//a/@href")

IMPORTXML(A2;B2)

Sintaxis

IMPORTXML(url; consulta_xpath)

url: URL de la página que se examinará, incluido el protocolo (por ejemplo, http://).

El valor de url debe ir entre comillas o ser una referencia a una celda que contenga el texto adecuado.

consulta_xpath: consulta XPath que se va a ejecutar en los datos estructurados.

Web Scraping

Es el proceso de extracción de datos de sitios web de manera automatizada.
Se realiza mediante el uso de herramientas y técnicas especiales para extraer información relevante de una página web y guardarla en un formato estructurado, como un archivo CSV o una base de datos.

Mozilla Firefox

1º Buscamos en el menú superior derecha, más herramientas y herramientas para desarrolladores, despues, vamos pulsando el elemento que queremos importar

2º pulsamos sobre el elemento con el botón derecho del ratón y en copiar, Xpath

3º Después lo pegamos en la fórmula importxml

https://www.aemet.es/es/eltiempo/prediccion/espana?a=pb
//*[@id="cabecera"]

=importxml(A1;A2)

Búsqueda de información en documentos XML: lenguajes de consulta y manipulación.

XPATH

La última versión de XPath es XPath 3.1, que fue publicada en 2017

1º Puede usarse en Java, JavaScript, Python, PHP, etc

2º se usa en XSLT

3º es un lenguaje de nodos.

XML

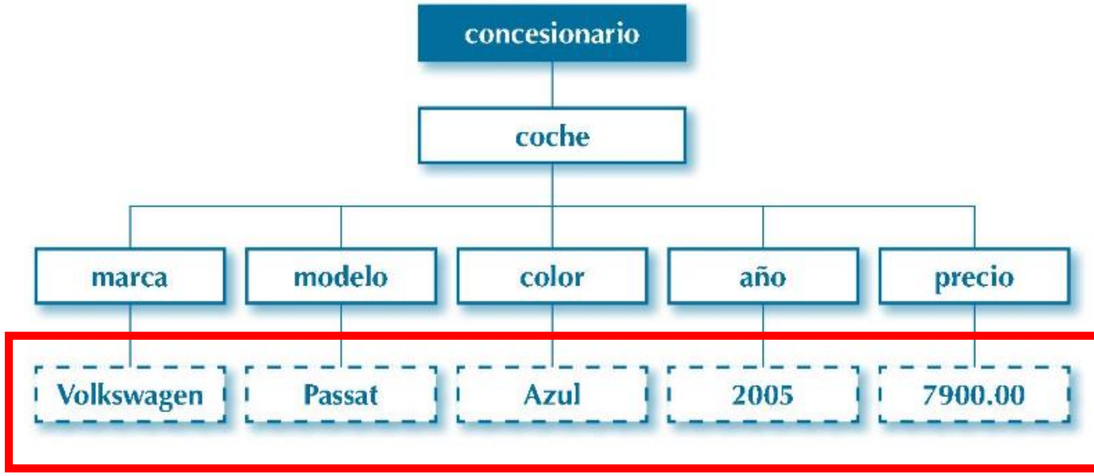
```xml
<?xml version="1.0" encoding="UTF-8"?>
<concesionario>
    <coche>
        <marca>Volkswagen</marca>
        <modelo>Passat</modelo>
        <color>Azul</color>
        <año>2005</año>
        <precio>7900.00</precio>
    </coche>
</concesionario>
```

nodo texto

nodo

nodo raíz

ARBOL DE NODOS



Valor atómico también puede ser también denominarse item o elemento.

XPATH

Que expresiones se usan por XPATH en la selección de nodos.

Expresión	Descripción
nombre_nodo	Selecciona todos los nodos con el nombre *nombre_nodo*.
/	Selecciona desde el nodo raíz.
//	Selecciona los nodos del documento que cumplan los criterios de selección desde el nodo actual sin importar dónde se encuentren.
.	Selecciona el nodo actual.
	Selecciona el nodo padre del nodo actual.
@	Selecciona atributos.

Ejemplo usando XPATH EN BASEX

XML

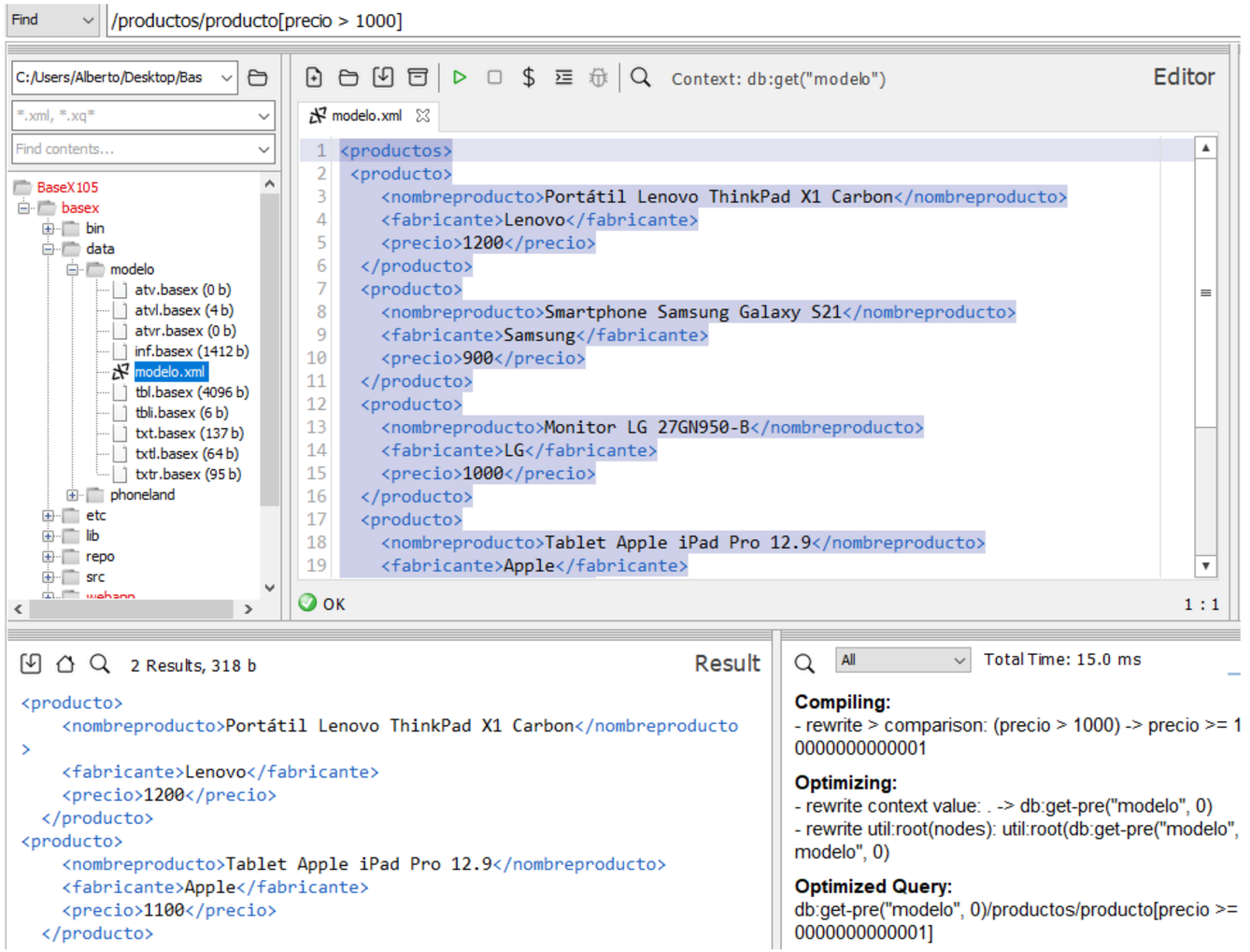
```
<productos>
 <producto>
   <nombreproducto>Portátil Lenovo ThinkPad X1 Carbon</nombreproducto>
   <fabricante>Lenovo</fabricante>
   <precio>1200</precio>
 </producto>
 <producto>
   <nombreproducto>Smartphone Samsung Galaxy S21</nombreproducto>
   <fabricante>Samsung</fabricante>
   <precio>900</precio>
 </producto>
 <producto>
   <nombreproducto>Monitor LG 27GN950-B</nombreproducto>
   <fabricante>LG</fabricante>
   <precio>1000</precio>
 </producto>
 <producto>
   <nombreproducto>Tablet Apple iPad Pro 12.9</nombreproducto>
   <fabricante>Apple</fabricante>
   <precio>1100</precio>
 </producto>
 <producto>
   <nombreproducto>Altavoz Sonos Beam</nombreproducto>
   <fabricante>Sonos</fabricante>
   <precio>400</precio>
 </producto>

</productos>
```

INSTRUCCIÓN XPATH

```
/productos/producto[precio > 1000]
```

RESULTADO



Ejemplo usando XPATH EN BASEX

INSTRUCCIÓN XPATH

Este XPath seleccionará todos los productos que sean fabricados por Apple o Samsung

/productos/producto[fabricante='Apple' or fabricante='Samsung’]

Buscar el precio de todos los productos que tengan un nombre de producto que contenga la palabra "Tablet":

/productos/producto[contains(nombreproducto, 'Tablet')]/precio

Buscar todos los nombres de los productos que tengan un precio menor a 500:

/productos/producto[precio<500]/nombreproducto

Para buscar los productos cuyo precio esté entre 500 y 1000 en XPath, se puede usar el siguiente código:

Combinar varias rutas

/productos/producto[precio>=500 and precio<=1000]/nombreproducto

Comodines

Comodín	Descripción
*	Selecciona cualquier nodo elemento.
@*	Selecciona cualquier nodo atributo.
node()	Selecciona cualquier nodo de cualquier clase.

Para combinar rutas XPath, se puede utilizar el operador | que permite seleccionar elementos que cumplen una de las dos condiciones. Por ejemplo, para seleccionar los productos con precio menor a 500 o mayor a 1000, se podría utilizar la siguiente ruta:

/productos/producto[precio<500]|/productos/producto[precio>1000]/nombreproducto

Uso de comodines ejemplo

Seleccionar todos los elementos y atributos de producto que tengan un precio mayor o igual a 1000:

/productos/producto[precio>=1000]/* | /productos/producto[precio>=1000]/@*

La expresión /productos/producto[precio>=1000]/* selecciona todos los elementos descendientes del elemento producto que tengan un precio mayor o igual a 1000, mientras que la expresión /productos/producto[precio>=1000]/@* selecciona todos los atributos del elemento producto que tengan un precio mayor o igual a 1000.

En conjunto, la expresión selecciona todos los elementos y atributos dentro de los elementos producto que tengan un precio mayor o igual a 1000.

Si pusiéramos **node(),** el uso de la función **node()** seleccionaría todos los nodos hijos del elemento producto que cumplen la condición de tener un precio mayor o igual a 1000, incluyendo tanto elementos como atributos. Entonces, la expresión quedaría como /productos/producto[precio>=1000]/node().

Prueba a realizar combinaciones

XQUERY

DIGI TECH

Instrucción	Para qué sirve	Ejemplo
for	Bucle para recorrer una secuencia de elementos	for $i in (1 to 5) return $i * 2 (devuelve la secuencia (2, 4, 6, 8, 10))
let	Asigna una variable para usarla en la consulta	let $x := 5 return $x * 2 (devuelve 10)
where	Filtra los resultados de una consulta	for $i in (1 to 10) where $i > 5 return $i (devuelve la secuencia (6, 7, 8, 9, 10))
if	Estructura condicional	if (2 + 2 = 4) then "Verdadero" else "Falso" (devuelve "Verdadero")
some	Verifica si algunos elementos de una secuencia cumplen una condición	some $i in (1 to 5) satisfies $i > 3 (devuelve true)
every	Verifica si todos los elementos de una secuencia cumplen una condición	every $i in (1 to 5) satisfies $i > 3 (devuelve false)
return	Devuelve los resultados de una consulta	for $i in (1 to 5) return $i * 2 (devuelve la secuencia (2, 4,

- Permite consultar y procesar información en formatos XML, incluyendo documentos XML y bases de datos XML.
- Permite seleccionar y filtrar información XML utilizando expresiones XPath.
- Permite combinar múltiples fuentes de información XML en una sola consulta.
- Permite transformar XML a XHTML
- Permite realizar operaciones de transformación de XML, como la conversión de formatos de fecha y hora, la agregación y agrupación de datos, y la generación de nuevos documentos XML.
- Permite trabajar con datos no XML, como archivos de texto y bases de datos relacionales, a través de la integración con otros lenguajes de programación como Java y JavaScript.
- Es compatible con múltiples plataformas y lenguajes de programación, lo que permite su integración en diversos sistemas y aplicaciones.
- Es un lenguaje declarativo, lo que significa que se centra en lo que se quiere obtener en lugar de en cómo obtenerlo, lo que facilita su aprendizaje y su uso para procesamiento de grandes volúmenes de datos.
- Es ampliamente utilizado en aplicaciones web y en la integración de sistemas, especialmente en el ámbito empresarial.

XQUERY: FUNCIONES

DIGI TECH

Función	Descripción	Ejemplo
fn:string()	Convierte un valor a una cadena de caracteres.	string(5) devuelve "5".
fn:substring(string, start, length?)	Devuelve una subcadena de una cadena dada. El tercer parámetro es opcional y especifica la longitud de la subcadena.	substring("Hola mundo", 2, 4) devuelve "ola ".
fn:concat(string1, string2, ...)	Concatena dos o más cadenas de caracteres.	concat("Hola", " ", "mundo") devuelve "Hola mundo".
fn:contains(string1, string2)	Devuelve true si la cadena string2 se encuentra dentro de string1.	contains("Hola mundo", "mundo") devuelve true.
fn:count(seq)	Devuelve el número de elementos en una secuencia dada.	count((1, 2, 3, 4, 5)) devuelve 5.
fn:sum(seq)	Devuelve la suma de los elementos en una secuencia dada.	sum((1, 2, 3, 4, 5)) devuelve 15.
fn:avg(seq)	Devuelve el promedio de los elementos en una secuencia dada.	avg((1, 2, 3, 4, 5)) devuelve 3.
fn:max(seq)	Devuelve el valor máximo en una secuencia dada.	max((1, 2, 3, 4, 5)) devuelve 5.
fn:min(seq)	Devuelve el valor mínimo en una secuencia dada.	min((1, 2, 3, 4, 5)) devuelve 1.
fn:distinct-values(seq)	Devuelve los valores únicos en una secuencia dada.	distinct-values((1, 2, 3, 3, 4, 5)) devuelve (1, 2, 3, 4, 5).
fn:empty(seq)	Devuelve true si la secuencia dada está vacía.	empty(()) devuelve true.
fn:not(boolean)	Devuelve el valor opuesto de un valor booleano dado.	not(true()) devuelve false.
fn:exists(seq)	Devuelve true si la secuencia dada no está vacía.	exists((1, 2, 3)) devuelve true.

1. **Almacenamiento XML nativo.**

EXIST DB

ExistDB es una base de datos NoSQL basada en documentos, específicamente diseñada para almacenar y gestionar documentos XML. A continuación se presentan algunas de sus principales características:

1.Almacenamiento de documentos: ExistDB se utiliza principalmente para almacenar y recuperar documentos XML, que pueden ser consultados y actualizados utilizando lenguajes de consulta como XQuery.

2.Escalabilidad: ExistDB está diseñada para manejar grandes volúmenes de datos y es altamente escalable. Puede ser utilizado en sistemas de cualquier tamaño, desde pequeñas aplicaciones hasta grandes proyectos empresariales.

3.Alta disponibilidad: ExistDB se puede configurar en una arquitectura de alta disponibilidad, lo que significa que puede estar disponible para su uso en todo momento, incluso en caso de fallos o interrupciones en el sistema.

4.Soporte para múltiples lenguajes: ExistDB admite una amplia variedad de lenguajes de programación, incluyendo XQuery, XSLT y JavaScript, lo que lo hace altamente flexible y adaptable a diferentes necesidades.

5.Interfaz de usuario web: ExistDB proporciona una interfaz de usuario web, llamada eXide, que facilita la creación, gestión y consulta de documentos XML y la realización de operaciones de mantenimiento y administración de la base de datos.

6.Integración con otros sistemas: ExistDB se integra fácilmente con otros sistemas y herramientas, lo que lo hace ideal para su uso en proyectos que requieren la integración de diferentes aplicaciones o plataformas.

7.Open source: ExistDB es una base de datos de código abierto, lo que significa que es gratuita y puede ser utilizada y modificada por cualquier persona con fines comerciales o no comerciales.

1. **Almacenamiento XML nativo.**

BASEX

1.Almacenamiento y procesamiento eficiente de grandes volúmenes de datos XML y de texto.

2.Funciones integradas y soporte completo para XQuery y XPath.

3.Escalabilidad horizontal y vertical para adaptarse a diferentes tamaños y requisitos de la aplicación.

4.Soporte para diferentes formatos de datos como XML, JSON, CSV y otros.

5.Funciones de procesamiento de texto avanzadas, como análisis y tokenización de texto, búsqueda de texto completo y clasificación.

6.Funciones de administración y gestión de bases de datos como copia de seguridad y restauración, monitorización de rendimiento y programación de tareas.

7.Soporte para varios lenguajes de programación como Java, Python, JavaScript y otros.

8.Flexibilidad y modularidad para adaptarse a diferentes requisitos de la aplicación y entornos de desarrollo.

9.Documentación y comunidad activa de usuarios y desarrolladores.

¿Cuando usar Basex y ExistDb?

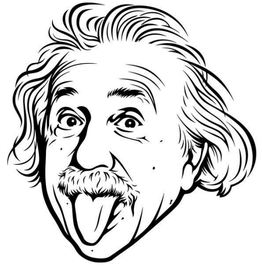
Si el proyecto requiere un **alto nivel de escalabilidad, ExistDB** sería la mejor opción. Porque es capaz de manejar grandes volúmenes de datos.

En el caso de que el **rendimiento** es más importanteuna consideración clave, BaseX sería la mejor opción ya que tiene una arquitectura más simple y se enfoca en la velocidad de procesamiento.

opción. Si la velocidad de procesamiento y el rendimiento son clave, BaseX podría ser la mejor opción

¿Si tengo bases de datos relacionales las puedo consultar en XQUERY y como?

Sí, es posible consultar bases de datos relacionales utilizando XQuery mediante el estándar XQuery for SQL. Este estándar permite a los usuarios expresar consultas en XQuery sobre bases de datos relacionales, utilizando una sintaxis similar a la de SQL.



AMPLIACIÓN COMO USAR XQUERY FOR SQL

DIGI TECH

Para utilizar XQuery for SQL, es necesario utilizar un conector JDBC (Java Database Connectivity) para establecer la conexión con la base de datos relacional y ejecutar consultas en XQuery utilizando las funciones y operadores específicos de XQuery for SQL.

Algunas bases de datos relacionales populares que soportan XQuery for SQL son Oracle, MySQL y PostgreSQL. Para utilizar XQuery for SQL en estas bases de datos, se pueden utilizar librerías JDBC como Saxon-JDBC, BaseX JDBC o exist-JDBC.

Ejemplo

```
xquery version "3.0";

import module namespace sql = "http://basex.org/modules/sql";

let $conn := sql:connect("jdbc:mysql://localhost:3306/phoneland", “root", “")
let $result := sql:execute($conn, "SELECT nombre, apellido FROM clientes WHERE ciudad= ‘Málaga'")
return $result
```

En este ejemplo, se establece una conexión a una base de datos MySQL llamada “phoneland" utilizando el conector JDBC correspondiente. Luego, se ejecuta una consulta SQL que selecciona los nombres y apellidos de los clientes que son de Málaga Finalmente, se devuelve el resultado de la consulta.

Trabajo de investigación

Investiga sobre 10 web diferentes a tu elección, y realiza web scraping con la función IMPORTXML. Abre una hoja de cálculo en Google Sheet y utilizando la función IMPORTXML, importa información xml el fichero de Excel súbelo después a la plataforma.

RECURSOS

XPATH

http://www.w3schools.com/xml/xpath_intro.asp

XQUERY

XQuery Tutorial (w3schools.com)

BASE X

Ejercicios XQuery #1 - ¡Primeros pasos con XQuery! – YouTube

EXISTDB

AD05_01 Instalación eXist db – YouTube

GRACIAS

Digitechfp: Alberto R,

Nombre ciudad
www.digitechfp.com